# The SCEPTICS Tool for Threat Analysis in Industrial Control Systems

Mihai Ordean
University of Birmingham
Birmingham, UK
M.Ordean@cs.bham.ac.uk

Richard J. Thomas*
University of Birmingham
Birmingham, UK
R.J.Thomas@cs.bham.ac.uk

Tom Chothia
University of Birmingham
Birmingham, UK
T.P.Chothia@cs.bham.ac.uk

## ABSTRACT

This paper presents a modelling tool for complex ICS systems that performs automatic threat analysis. The aim of this tool is to discover and enumerate complex attack paths through a system, making it possible to assess the risk to particular assets and test new strategies for mitigating risks, as well as helping ICS owners to understand their systems. Current tools allow system owners to identify maximal strategies to protect their systems against particular attacks, however, they do not allow the asset owner to discover threat propagation and rank threats to their system, or they require the system owner to determine the probability risk themselves. We employ probabilistic analysis using the CVSS framework, allowing system owners to concentrate on defining their architectures, rather than deriving potentially incorrect values of risk to their system. The results of the tool can be used to provide assurance and prioritise security improvements to a system. We provide an extensive example of our tool in use, modelling the security of the ERTMS Rail Signalling Standards and on-board train systems.

## CCS CONCEPTS

**•Security and privacy → Formal security models; Systems security;**

## 1 INTRODUCTION

In IoT and ICS, there is an emphasis on functionality and safety over security. This presents an issue as the system ages, where security was not a primary requirement. As an example, the train to trackside communications protocols in the European Rail Traffic Management System (ERTMS) employ old ciphers for confidentiality (A5/1 which dates to 1987) and integrity (EuroRadio, proposed in 1997 [3]). Today, vulnerabilities exist to these protocols [8, 23], where system owners now recognise the need for security, and the requirement to appraise their exposure to threats.

Modelling tools are already used by engineers to rationalise safety assumptions and validate the design of their systems, where the same technique could be used to assess the security of a given architecture. Tools are available today which allow an asset owner to model the security of their system but have limitations, for example requiring the asset owner to express the risk profile of their systems. This requirement can be prone to human error, either due to a misunderstanding, or the domain and subject-relevant knowledge not being captured as part of the overall system profiling, e.g. using the Altran 'REVEAL' methodology [13].

A minimalistic security assessment should address the following 5 questions: (1) what do I want to protect?, (2) who do I want to protect it from?, (3) how likely is it that an element needs protection?, (4) how severe are the consequences in the event of a compromise?, and (5) what is the cost of preventing compromise?

Risk and hazard analysis, however, is a subjective process which requires a lot of understanding about the components of a system, especially when these components are interconnected. Threats to some components such as safety critical elements are deemed unacceptable by the asset owners, no matter what the risk, as the mere presence of the threat at any likelihood could endanger human lives. The evaluation of risk, probability and impact needs to be carefully balanced, given that a high risk may have a negligible consequence, and, therefore be tolerated and accepted by the asset owner. As an example, let us consider a national infrastructure which operates with a vulnerable communications protocol running between nodes. An attacker who is able to compromise this protocol could potentially have unrestricted access to the nodes. More concretely in ERTMS, from the perspective of a train using GSM-R, there would be the potential for a disruptive effect by an adversary overloading the EuroRadio layer with messages that cannot be processed, either due to poor formatting, or an invalid MAC, which could trigger the session to be terminated. Conversely, an air-gapped interface between the train control systems and, say, a passenger seat reservation system would have little impact on the safe operation of the train but would have a disruptive effect on the passengers. These risks should be reliably modelled and captured to ensure they are understood and do not affect compliance, e.g. with the NIS Directive thresholds.

Critical National Infrastructure and ICS system owners are now faced with requirements to comply with the EU Network and Information Security (NIS) Directive as well as implementing ISO/IEC 62443, where there is a large knowledge, experience and skills gap to reach compliance. As a result, some organisations are unable to assure their infrastructure, as they do not understand the risks that exist in their architectures. Malware, for example Stuxnet, leveraged the fact that systems that were meant to be airgapped had data transferred between each other via USB, allowing its propagation. A more modern variant, BlackEnergy [20] uses spearphishing to affect a management workstation to make it a pivot point, then affecting other ICS components. These are but a few examples of threats as the result of increased interconnectivity that were not realised, nor through the connectivity of these systems that the risk was identified, or truly appreciated and evaluated. Today, in the mainstream media, the work by INSINIA[1] highlights the point of safety over security when systems were developed, and

*Equal contribution to the first author

[1]http://www.theregister.co.uk/2018/06/18/physically_hacking_scada_infosec/

increased connectivity, e.g. remote monitoring were convenient but the security foresight was lacking. Methodologies, for example, one presented in [10] considers the threats to the railways, and defines a process to identify security risks to infrastructure. Using frameworks like this, automated tools for use by ICS owners can be created to support initial analysis by system owners.

In the tool[2] we present in this paper, we model components of a given system as a graph and the data flows between these components. We also consider how data changes as it passes through various systems, e.g. location data may be converted into some authenticated data. It is important to consider this type of data conversion, as it may be the case that unauthenticated data may become safety-critical command and control data after passing through a number of nodes. Moreover, tracking the data types (a representation of the data that is transferred between nodes) and link types (representing the properties of the point-to-point connection between nodes) allows the asset owner to generate a representative model of their system and simulate different adversary models (e.g. an inside threat, a compromised workstation or an attack taking place to the signalling centre).

Using this foundation, we can specify key assets to assess as 'targets' to the attacker, which can be modelled as having entry points at any point in the model with various capabilities. The tool, developed with a Microsoft Visio frontend, is able to take a graphical model created by the asset owner, representing the probability space and supplemented with XML-like attributes of nodes and edges in the model and subsequently find paths through the model between assets and assess the compositional security that is offered. Using this model and path-finding capability, the tool returns attack paths and the probability of an exploit being successful on that path, highlighting some potentially interesting paths that were not previously considered or identified to the asset owner.

In this paper, we develop a comprehensive model of ERTMS and an example train bus, created by examining the set of standards we were given access to and interviews with groups of engineering-focused (non-security) rail experts. We will demonstrate the use of the SCEPTICS tool on this reference model, highlighting some previously undisclosed, perhaps unsurprising attack vectors that exist within the system which, to our knowledge, is the first time these attack vectors have been output and expressed using an automated tool.

**Contributions.** In summary our contributions are as follows:

(1) **Security benchmarking of complex systems:** Our tool leverages the Common Vulnerability Scoring System (CVSS) to derive the security profile of individual system components and links, and uses the inclusion-exclusion principle to determine the security of a group of elements. As the tool is dependent on the design of the system, comparisons may be drawn by the system owner about the security of these systems.
(2) **Accurate, automated security assessments:** The main objective of our tool is to provide meaningful assessments of the security of ICS environments that allow comparisons between versions of a system or multiple systems. The tool is aimed at non-security individuals who are required to understand the security implications that occur when modifications are made, or, for example, not updating firmware, but does not remove the requirement for expert security assessments.
(3) **Supplementing security analysis:** Our tool assists the process of security assessments by highlighting 'interesting' or critical specific system components which have a higher impact on the overall security of a given system. The analysis it carries out allows system owners to prioritise improvements to their system to reduce their exposure. It does not, however, remove the need for expert security assessments, which may highlight details which cannot be captured through the asset owner's lack of detailed security knowledge.
(4) **Allowing ICS System Owners to simulate and experiment on the security of their system:** The tool can be used to experiment and assess various optimisations and strategies to improve the security of an architecture before major modifications or cost is incurred, where the asset owner can compare the various models generated. This has the benefit of allowing the asset owner to assess which strategies are the most economical and improve the security of the system overall.
(5) **Enabling propagative security analysis of exploitation:** Our tool is is able to consider 'bridges' between multiple data types, i.e. where data is transformed, allowing the asset owner to simulate a 'cyber kill chain' and follow the path an attack may follow.

## 1.1 Overview

In this paper, we propose a new tool that allows system engineers and non-security experts to assess the security of these types of systems starting from a generalised system architecture. We begin with the simplified diagram presented in Fig. 1, an example of several interconnected components that exist in rail, based on Fig. 2, a much more complex, interconnected system.

Our tool models *system components and data flows* as a directed graph, i.e. a collection of nodes connected by directed edges, where the nodes represent the system components and the edges represent the data flows between them. For the modeller, a Visio graph is created which the tool transforms into an XML-like representation which is then computed into a mathematical model. In the mathematical model, data types and link types (specified in the Visio model by the asset owner) can be used to constrain the search space to a specific type of data or communication link that may be compromised.

The modeller can define all the properties and specifications of the system components that will be used. The tool uses the following three basic concepts: (1) system components, (2) data profiles and (3) process operations to automatically asses the security of any modelled system.

*System components* are the actual physical components of the network. These can be entities such as embedded system boards,

[2]A copy of the full model presented in this paper and the SCEPTICS tool, with accompanying documentation, is available to download from https://github.com/sceptics-tool/sceptics-tool.

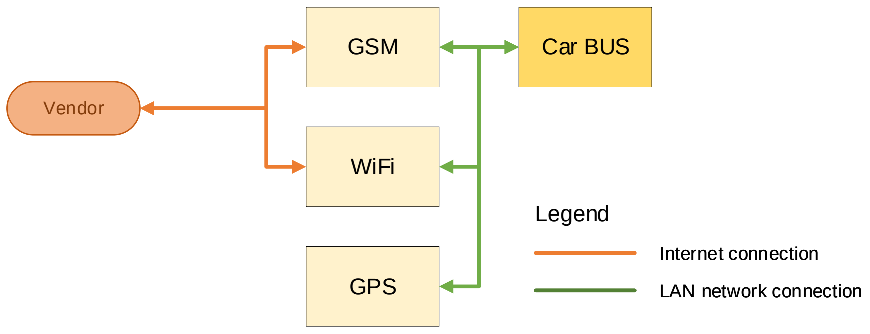

**Figure 1: Toy example model based on Figure 2. The *GSM, WiFi* and *GPS* nodes refer to the appropriate equipment used to provide a service, where the link between the *WiFi* node and *Vendor* represents a backend connection (e.g. via cellular technologies).**

networking equipment, or physical sensors, but also more abstract components such as BUS-es, network interfaces or even software. For example, in Fig. 1, we show two groups of nodes: train components (yellow and dark yellow nodes), and components outside the ERTMS system (orange nodes).

Additionally, we are also modelling the types of links such as local LAN data links (green) or external wide-area data links (yellow). Our tool also supports various particularities for these links such as the support medium (e.g. wired, wireless, electrical) which are shown in Fig. 2.

*Profiles* are numerical values that describe either some individual aspect of the security of a system component (e.g. tampering with instructions sent from the vendor's network to the the train) or an overall aggregated value for a component. These numerical values are used by the process operation to derive the aggregate security characteristic of the analysed model. Out tool uses the Common Vulnerability Scoring System (CVSS) to establish these values as objectively as possible. However, we acknowledge that various (subjective) ways of establishing these are also possible, as such we also allow the modeller to custom-define values. The means by which these values are seeded into the model is discussed in Section 3.1.

The *process operation* is the method used to combine and aggregate security profiles of individual components (e.g. WiFi, Car BUS, GSM) to assess the likelihood of events such as: *What is the likelihood that tampering with the train WiFi connection can be exploited in a way that it could lead to two trains crashing into each other?*

We achieve this by using the inclusion-exclusion principle [4], to combine the individual profile values associated to each system component in the model and compute overall values for specific groups of components as required by the analysis performed. This allows us to accurately and exactly compute probabilities of events without using a sampling model.

Another benefit to this approach is the ability to detail or contract a model according to the needs of the asset owner (e.g. cases in which details are not interesting or available for the analysis or when the sub-model has been previously analysed and its details have not changed). For example, Fig. 2 presents a relatively detailed version of a "train car", based on [25, 26]. If the modeller is not interested in the security of individual train component nodes, a combined profile can be computed (or assigned), and the train component nodes can be replaced with a single "train car" node. This approach can be applied for any group of nodes that can be represented as a complete model.

Similarly, the opposite approach can also be taken where components such as the Radio Block Center (RBC) node in Fig. 2) can be expanded and analysed as a group of individual subcomponents. The RBC is responsible for interfacing between the rail interlocking system, a system that prevents conflicting movements in railway signalling, and issuing command and control messages to trains in its region of control.

## 2 RELATED WORK

Modelling of complex systems is prevalent in a number of sectors, and is now progressing to its application in information security. With the EU Network and Information Systems (NIS) Directive deadline for compliance looming, system owners are looking for new solutions that allow them to assess and appreciate the security of their infrastructure. It is important that such verification and modelling takes place during the design phase as well as during the operation of these systems [1].

For Industrial Control Systems, Khan et al. [16], the STRIDE (Spoofing, Tampering, Repudiation, Information disclosure, Denial of Service, Elevation of Privilege) framework is applied to a small synchronous island model. In this framework, five key steps are carried out, namely reducing the system into underlying components, creating a data flow diagram of the system under assessment, analysis of threats to this diagram, identification of vulnerabilities and planning mitigations to those vulnerabilities identified. In this methodology, threat consequences are identified, where the individual components are matched to threats under the STRIDE phases. From the outset, this framework operates based on a snapshot analysis of a system, but does not consider propagative effects of any vulnerabilities identified to the system under assessment.

An alternative way of modelling is proposed by Fielder et al. [11] using particle swarm optimisation in a model of a system under assessment to identify ways in which the best appropriate defences can be applied to a system, which have a measurable impact. In this work, simulations of a model are carried out to inform such strategy development, however it is not clear how inputs to the system are handled. It however provides an insight into attack and defence strategies that could be considered in complex systems to deliver optimal security, even if at a constrained cost. In comparison to using NetLogo 'agents' to represent the input, Loddertedt et al. [22] use UML-like language to represent their model as input for their SecureUML tool, which models an access control system, generating a supporting infrastructure which conforms to the specifications that are provided by the system owner.

Whilst UML and agent-based modelling are potential approaches which an asset owner may use, graph-based risk modelling has been applied to cyberphysical settings previously where, as an example, Santini et al. use Evidence Theory as a way to express the risk that exists in complex, interconnected systems [27]. What is key here is that Santini et al. allude that there must be some union between the domain and subject knowledge, providing a contextual point of reference for the values presented to the asset owner, supporting the development and maintenance of the model. Lautier et al. in [19] apply graph theory to an alternative context, specifically the finance sector to assess the risk to markets, where their framework is able to search for, and find, the most probable and shortest path that would result in fluctuations in stocks. Whilst this framework could be applied to a cyber-physical system setting, the framework has the risk of not highlighting alternative paths of risk, where an attacker would chain a set of vulnerabilities together which are simpler to exploit, although with a longer path which the framework proposed by Lautier et al. would not find.

When considering the integration of CVSS metrics into attack graphs, there are a number of different approaches which do not necessarily work for a probabilistic model. The CVSS metric itself is only a 'snapshot-in-time' view of the system under assessment, where the interdependency between components are not captured, preventing a rich analysis of the system architecture as a whole. As an example, Cheng et al. [7] provide an insight on how a CVSS score can be transformed into a probabilistic measure of an attack being successful. This work however lacks the ability to capture dependencies between components, which is achieved in our modelling tool, where we are able to traverse the graph from a selection of starting nodes to calculate values across attack paths.

In a different context, Ioannidis et al. [15] use modelling for security operations to determine the trade-offs that exist for patch-management and determining the optimal strategy for patching without affecting overall system security. Beautement et al. in [2] also apply this approach, adapting the the security requirements of a system to determine appropriate strategies. These trade-offs are decisions that ultimately may have to be made as part of the modelling process when potential issues are identified. Anderson [1] notes the problem that trade-offs present and how the balance between the financial investment to mitigate or reduce the risk can be struck against its likelihood and impact.

An alternative way that asset owners can determine the appropriate action to take (if they have an established and well-understood adversary model) is through the use of Attack Trees [1]. Kordy et al. [17] describe an extension to this technique using Attack-Defence Trees, where in addition to considering the time and expense required to 'defeat' a given node, the tree is supplemented with 'defender' nodes, each with thier own defined costs and contextual information. The output of the trees allow the asset owner to quantify and balance the 'cost' of an attack against the cost of implementing defences. Byres et al. [5] assess the use of Attack Trees in SCADA applications, in particular identifying that asset owners may become consumed with resolving the potential vulnerability of a given node, when the relative risk of exploitation is low. In their work, the authors outline some example trees which are assessed for the relative difficulty that the adversary would face, the severity of exploitation and the likelihood to which an adversary would be detected. An extension of Attack-Defence trees by the SPARKS Horizon 2020 project, HYRIM [14], applies the technique to smart-grids, where pre-existing vulnerabilities are taken into account within the architecture. These may be generalised threats (e.g. lack of authentication) or link directly to a CVE that affects a particular component. What the authors note is that there is a risk that the tree can become too detailed and large, introducing a degree of redundancy. There is, therefore, a need for a deep understanding of the attacks that may arise, where a methodology is proposed to establish 'good' attack graphs.

Both approaches, however rely on the system owner having prior knowledge of the potential attack vectors that could exist in their system architectures, a requirement which may require many iterative trees, where errors may be made. Unlike attack trees, a model-based approach can view the system as a whole, rather than as a set of siloed components. Moreover, security is quantified as an estimated cost metric, which may be incorrect where security profiles can be used instead, mapping to the goal-based approach taken by attack-defence trees.

Shostack [28] reviews a number of the previously-discussed processes, including STRIDE and attack trees and how they can be managed and addressed. In particular, considerations as to how to deal with the potential type of trade-offs that may arise as a result of the modelling process, e.g. whether the risk identified should be accepted, mitigated, avoided or transferred.

In a number of sectors which use cyber-physical systems, in particular in rail, risk assessments of safety-critical systems are undertaken using safety modelling. These assessments define the possible hazardous events and scenarios that could lead to a threat to life. From this, the causal and consequential analysis is carried out where the risk is evaluated with actions taken to mitigate, reduce or remove that risk factor. This approach is typically safety-driven but can be adapted for security. This, however, would suffer from the same flaws that exist for attack-defece trees, where some definitive metric is required to quantify the security impact and investment required to compromise an asset. Matulevičius [24] proposes an alternative approach using the ISSRM security metric, which factors in the likelihood and vulnerability level (the ease to exploit a component). Vulnerabilities can be seen to influence the likelihood of exploitation, but is a more useful metric, where more attractive targets may exist for an adversary, where the capabilities of the 'threat agent' need to match the real-world, similar to the method applied by Fielder et al. [11].

Li and Hankin [21] extend the work of Fielder et al. to define a metric that measures the tolerance of a system to zero-day attacks. The authors employ Bayesian networks to model a simple ICS environment, observing the effects of applying controls (e.g. deploying firewalls or anti-virus solutions) to the system. In their model, they assess the propagative effect of zero-days over time, allowing asset owners to determine which combination of controls provides maximal defence against such vulnerabilities, an approach similar to that of the HYRIM project, where a combination of both approaches cold be effective. Again, one requirement is that the asset owner has to carefully control the threshold metrics, where its accuracy is critical.

One feature that is common to the aforementioned work is that there is an assumption of single dataflows within a given system. This, however, is not typically the case, where our tool is able to capture the notion of data types, e.g. authenticated data and location data, where we can 'bridge' between data types, following the transformation of data throughout the system. This enhances an asset owner's ability to model the system under assessment and understand how an attack can propagate due to this data transformation.

In the United Kingdom, the National Cyber Security Centre (NCSC) introduced the Cyber Assessment Framework (CAF)[3] as a means of bridging the gap for ICS owners to be able to assess their level of compliance with the EU NIS Directive. The CAF is broken down into NIS objectives, aligned with the NIST framework, where asset owners provide evidential 'indicators of good practice', similar to ISO 27001. This framework, however requires a level of security knowledge to be able to confidently assess compliance with each objective. No automated tool exists where you can demonstrate efficacy and compliance to the principles, which would allow for faster, more accurate assessments. It also is limited in its efficacy without an automated tool to ensure compliance, allowing for faster, more accurate and non-subjective assessments.

## 3 THE MODELLING TOOL

In this section, we give a detailed description of our tool starting with the grammar, followed by our probability based computation method, and then by the way we assign security values to the system components. We will then present specific details related to our implementation, focusing on important design choices.

### 3.1 The Modelling Tool Input Grammar

In order to assess the security of a system, our tool requires three inputs: (1) a *system graph* that describes the individual components and data flows of a system, and the way these interact with each other, (2) an *adversarial model* which describes the capabilities of the attacker, and (3) a *list of assets* which are the attacker-targeted components of the system. Using these three inputs, our tool is able to identify the most likely paths that particular attackers might use to attack the assets and so identify the most vulnerable elements in the system and assess the effect of mitigations. The detail and type of analysis is controlled through the adversarial model and the list of assets. Formally, the grammar of the inputs is:

| | | |
|---|---|---|
| ⟨*system graph*⟩ | ::= | List⟨*node*⟩ List⟨*edge*⟩ |
| ⟨*assets*⟩ | ::= | List⟨*asset*⟩ |
| ⟨*adversarial model*⟩ | ::= | List⟨*adversary*⟩ |

**The system graph.** The system graph is structured as a directed graph that describes the architecture of the physical system to be analysed by the tool. The individual components of the physical system are modelled as the nodes of the graph and the connections between them are represented as edges. For example, in the rail diagram shown in Fig. 1, *GSM* and *WiFi* components have been modelled as nodes. The main bus, i.e. *Car BUS*, has also been modelled as a node, shown differently to other nodes, demonstrating how asset owners can define components as granular or generic, as required. As shown in Fig. 2, the *Car BUS* is exploded into its constituent components. This allows the modeller to capture the shared security features of this bus. Other internal connections like the one between the *GSM* and the *Car BUS* are modelled as edges which enable a finer grained security assessment particular to the link in question.

Each node in the graph is identified by a *node identifier* and also has a descriptive *name*. Additionally, nodes have a list of labels describing supported *data types*. The *data types* can refer to standardised elements like protocols (e.g. NTP, UDP) or they can be custom defined (e.g. geographical data, authenticated data). Nodes also supports bridge entries which describe the capabilities of a node to convert data from one label to another, e.g. from location data to authenticated data. Each node has a data profile, *DP*, associated to them. *DP*s have a *data type*, a *link type* (a label similar to *data type* meant to describe the connection medium, e.g. Ethernet, electrical or radio), and a CVSS profile, *CP*, which describes the security characteristics of the link. Data types represent the property of traffic transferred between nodes, whereas link types represent the properties of the point-to-point connection. We discuss the CVSS profile in Section 3.1. The grammar of a node is as follows:

| | | |
|---|---|---|
| ⟨*node*⟩ | ::= | "`id:`⟨*node id*⟩ `name:`⟨*name*⟩ `data_types:`List⟨*data type*⟩ `bridges:`List⟨*bridge*⟩ `data_profile:`⟨*DP*⟩ |
| ⟨*bridge*⟩ | ::= | ⟨*data type*⟩`:`⟨*data type*⟩ |
| ⟨*DP*⟩ | ::= | `data_type:`⟨*data type*⟩ `link_type:`⟨*link_type*⟩ `CP:`⟨*CP*⟩ \| `data_type:`⟨*data type*⟩ `CP:`⟨*CP*⟩ \| `link_type:`⟨*link type*⟩ `CP:`⟨*CP*⟩ \| `CP:`⟨*CP*⟩ |

Connections between system components are modelled as edges, as such, an edge is uniquely defined by the pair of nodes it links. Their grammar is similar to that of the nodes i.e. edges have one or more data profiles, *DP*, associated to them.

| | | |
|---|---|---|
| ⟨*edge*⟩ | ::= | `id:`⟨*node id*⟩ `id:`⟨*node id*⟩ `data_profile:`⟨*DP*⟩ |

For instance, the model of our toy example, given in Fig. 1 would be written as:

| | | |
|---|---|---|
| node1 | ::= | `id:`nGSM `name:`GSM `data_types:`[LAN net. con., WAN net. con.] `bridge:`[bridge1] `data_profile:`dpn1 |
| bridge1 | ::= | LAN net. con.`:`WAN net. con. |
| dpn1 | ::= | `CP:`0.381 |
| node2 | ::= | `id:`nCB `name:`Car BUS `data_types:`[LAN net. con.] `data_profile:`dpn2 |
| dpn2 | ::= | `CP:`0.972 |
| edge1 | ::= | `id:`nGSM `id:`nCB `data_profile:`dpe1 |
| dpe1 | ::= | `data_type:`LAN net. con. `data_link:`Ethernet `CP:`0.200 |

**Adversarial model.** The adversarial model input describes the capabilities of the adversary. This input is important because it allows

[3]https://www.ncsc.gov.uk/guidance/nis-directive-cyber-assessment-framework

the security evaluation to be restricted to meaningful adversaries. As such, each adversary can contain lists describing attack starting nodes from the graph, i.e. *entry nodes*, and/or exploitable *data types* and *link types*. The grammar for defining an adversary is as follows.

$\langle adversary \rangle$ ::= `entry_nodes`:List$\langle$*entry nodes*$\rangle$
`data_types`:List$\langle$*data types*$\rangle$
`link_types`:List$\langle$*link types*$\rangle$

An adversary example is:

adversary1 ::= `entry_nodes`:[$\epsilon$]
`data_types`:[WAN net. con.]
`link_types`:[Wireless]

The notation above models an adversary that is able to compromise any *WAN* network connections or *Wireless* communication links giving it access. Based on Fig 1, *adversary1* is composed of the *WiFi* and the *GSM* controller nodes as they both have a wireless connection interface. These represent the entry points into the system. An equivalent notation for *adversary1* would have been:

adversary1 ::= `entry_nodes`:[WiFi, GSM] `data_types`:[$\epsilon$]
`link_types`:[$\epsilon$]

As previously stated, asset owners can choose whether to generalise data types and link types, where the asset owner may use *WPA2* as a link type instead to represent WPA2 encrypted transmission, typically used on WiFi networks.

**Asset list.** The asset model input is similar in purpose to the adversary model, with the important difference that it focusses the analysis on the system components which are considered to be the target of an attack, i.e. the list of assets that the asset owner may consider critical or may want to understand if attacks can reach those assets. As an example, the ERTMS system owner might want to evaluate the ways in which the train *Car BUS* can be compromised. The asset can also be further restricted to specific data types handled by that node by defining a *data types* list. The grammar for the asset is given below.

$\langle asset \rangle$ ::= `id`:$\langle$*node id*$\rangle$ `data_types`:List$\langle$*data types*$\rangle$

In the following, we give examples of an unrestricted and a restricted asset from Fig. 1.

asset1 ::= `id`:nCB
asset2 ::= `id`:nCB `data_types`:[LAN net. con.]

**CVSS security profiles.** Defining the security of system components has major implications on how attack vectors, i.e. the node paths discovered by the tool, are ranked and the relationships between them. Furthermore, requiring system owners or modellers to specify the security of their assets (especially in a numeric format) is also unreasonable because they often lack the security expertise and/or they can over- or under-estimate how secure (or insecure) these assets are. For existing modelling tools, you can carry out a targeted analysis of systems, where the current state-of-the-art in modelling requires a high degree of manual effort and expertise to determine these specific values.

To address these issues, we propose a way to integrate the Common Vulnerability Scoring System (CVSS) framework[4] into our tool, and use it to uniformly generate and assign security profiles, i.e. *CP* values, to individual system components. The CVSS metric is particularly suited for this task because it represents the standard way of determining vulnerability scores using CVEs. Additionally, CVSS metrics also include information related to the severity of a vulnerability and other contextual information, such as the environmental impact or difficulty of fixing the vulnerability which allow for a more consistent granularity.

A CVSS vector is composed of three parts: the *base score*, *temporal score* and *environmental score*. The *base score* takes into account how the component may be attacked, e.g. via a network or physically, the complexity required for an attack, the privileges required and whether a user has to perform any actions to allow an attack to be successful. The impact on confidentiality, integrity and availability (CIA) is included in this score, as is the potential of an attacker to gain 'enhanced' privileges by compromising that component. The *temporal score* looks at the current security state of the component, e.g. if there are exploits in the wild, or if there are theoretical attacks, and how these threats may be remediated, if anything, and how confident we are in the literature and reports about the exploitability of that component. Finally, the *environmental score* evaluates how the compromise of the component will affect neighbouring, linked components, and what its requirements are and the impact if compromised on these requirements.

In general, CVSS vectors are converted to numerical values $i \in \{0, \ldots, 10\}$, using the equations defined in the CVSS standard [12] (we include the relevant equations in Appendix A). Our tool, however, uses a probabilistic computational model to rank and compute security values for groups of components and, thus, requires the CVSS values, $CP \in [0, 1]$. Additionally, our modelling method only needs the probability of success for an attack (i.e. how exploitable it is), so we transform the *base score* of the CVSS as follows:

$$CP = \frac{ESC - 0.121}{3.887}, \tag{1}$$

where *ESC* is the exploitability score derived from the CVSS vector value as:

$$\begin{aligned} ESC = 8.22 \times AttackVector \times AttackComplexity \times \\ \times PrivilegeRequired \times UserInteraction \end{aligned} \tag{2}$$

In Equation 1, we first scale the CVSS scores by removing the minimum interval of the ESC score (given in Equation 2) and then dividing it by the maximum interval of the ESC score to scale the value to a number between 0 and 1. Equation 2 is used in the CVSS framework as part of the calculation of the base score, where it specifically focusses on the exploitability of the component under assessment. For the purposes of our calculations, we do not consider the impact upon confidentiality, integrity and availability, thus, we only use the ESC component of the base score. Now, as we observe above, this value does not accurately reflect the likelihood that an exploitation attempt succeeds, where the ESC value is scaled using Equation 1 to give the final probabilistic value, which we use in our calculations.

**CVSS profile example.** Let us consider the GSM component in Fig. 1. This provides train to trackside communications, as well as a data link for other services on the train, and relies on the weak A5/1 encryption scheme. We model its CVSS vector and define it

[4]https://www.first.org/cvss/specification-document

as:

$$CVSS_{GSM}{=}AV{:}A/AC{:}H/PR{:}N/UI{:}N/S{:}C/C{:}H/I{:}H/A{:}H/$$
$$E{:}F/RL{:}U/RC{:}C/CR{:}L/IR{:}H/AR{:}H/MAV{:}N/$$
$$AC{:}H/MPR{:}N/MUI{:}N/MS{:}X/MC{:}L/MI{:}H/MA{:}H$$

This vector describes the security parameters of the GSM-R node such as the requirement of the attacker to have 'adjacent access' (AV:A), that is using a component of the stack (radio) to gain access to endpoints. That said, gaining access to the down-link (from mast to device) is trivial, however, affecting the up-link (device to mast) is complex due to the time-sensitive nature of GSM (AC:H), coupled with finding the precise timestamp and frequency being used by the train. For the remaining components of the CVSS vector, we will provide highlights of the vectors. For the temporal score, we acknowledged functional means to compromise the down-link (E:F) as rainbow tables for the A5/1 cipher is available, and it is currently unclear what remedial action is (RL:U) as the cipher is used across the world for GSM-R, and may require significant reimplementation and cost. We also put confidence in this vector (RC:C) based on tutorials and the outline work in [8] which shows how trivial it is to capture from the down-link.

The full list of parameters are available online[5]. By applying Equation 2 and Table 1, we compute $ESC = 1.6$. Using Equation 1 we can then compute the value corresponding to $CVSS_{GSM}$, i.e. $CP_{GSM} = 0.3805$. This value can be interpreted as the probability that an attacker can compromise or affect the GSM link, which would then have a propagative effect for dependent systems.

| Metric | Metric Value | Numerical Values |
|---|---|---|
| (Modified) Attack Vector | Network | 0.85 |
| | Adjacent Network | 0.62 |
| | Local | 0.55 |
| | Physical | 0.2 |
| (Modified) Attack Complexity | Low | 0.77 |
| | High | 0.44 |
| (Modified) Privilege Required | None | 0.85 |
| | Low | 0.62 (0.68 if Scope/ Modified Scope is changed) |
| | High | 0.27 (0.50 if Scope/ Modified Scope is changed) |
| (Modified) User Interaction | None | 0.85 |
| | Required | 0.62 |

**Table 1: CVSS Metric Values used by the Tool**

The attack path considers these dependent systems, showing how an attacker can leverage one weak component to cause some negative effect. An example attack path is the event of the GSM base station becoming compromised, affecting the onboard GSM system (antennae, modem and equipment) to the European Vital Computer (EVC), a system located onboard the train which processes messages and actions to/from the RBC over GSM-R, where a denial of service would lead to the train stopping. If an attacker were able to inject into GSM, a potential path would go from GSM, through the GSM network infrastructure to the RBC, where it could be given a false report or have messages replayed.

[5]https://www.first.org/cvss/cvss-v30-specification-v1.8.pdf

**Custom-defined security profiles.** Although using CVSS vectors reduces the likelihood of inconsistent security profiles being applied to components, it does not always match what can be discovered through an assessment of domain and subject-specific knowledge of a system, where there may be some mitigating factor which the CVSS vector cannot take into account. As such, we allow modellers to override the security profiles with correctly formatted user-defined values, $CP \in [0, 1]$.

The added benefit of this is that it enables modellers to run security simulations using the security profiles of chosen components as variables.

## 3.2 Definitions and Computational Model

In this section, we describe how attack vectors are modelled, followed by the method used to compute the security values for these attack vectors.

Let us begin by defining the system graph input from the the previous section (Section 3.1) formally as a directed graph $G(N, E)$, where $N$ is the set of all nodes (system components), and $E$ is the set of all edges (connections) in $G$.

Up to this point, we have a formal directed graph which needs to be transformed into a mathematical model, where only nodes are considered. We map our XML models onto this graph, where each link explicitly is converted into a node. Where a node handles more than one data type, it is split into individual nodes representing each data type, with the CVSS Profile/probability value replicated to the new nodes. Where a bridge exists between data types, links between these two nodes are created representing the conversion of data types within the node. What this achieves is a simple, but expressive mathematical model that can be computed upon. As an example, a node which bridges sensor and control data would be split into two nodes, one responsible for sensor data, the other for control data with a link between them. If it did not bridge between these two data types, no link between the two nodes would be created during the transformation step.

We consider a probability space $(\Omega, \mathcal{F}, P)$, where the sample space is $\Omega = \mathbb{P}(E)$ the power set of edges in $G(N, E)$; the event space is $\mathcal{F} = \mathbb{P}(\Omega)$; and $P$ is a probability function $P : \mathcal{F} \rightarrow [0, 1]$ that gives us the probability of a particular edge "existing", which in our model corresponds to the edge being exploitable by the attacker. Using the formalism above, an *attack vector* starting from a system component $n_s$ and targeting an *asset* $n_e$ is a *path* from $n_s$ to $n_e$ through the graph $G(N, E)$, defined as:

*Definition 3.1 (Path).* A path $\rho(n_s, n_e)$ in graph $G(N, E)$ is a sequence of unique edges $e_i \in E, i = 0 \dots n$ where: (*i*) the start of the path is given by the start node $n_s$ of $e_0$, (*ii*) the end is given by the end node $n_e$ of $e_n$, and (*iii*) for each pair of consecutive edges $(e_i, e_{i+1})$ the end node of $e_i$ is equal to the start node of $e_{i+1}$.

A path from the node $e_s$ to $e_n$ represents an attack which starts at $e_s$, following links and passing through intermediary nodes, reaching $e_n$. The end of the path ($e_n$) is an asset that an attack may take place, where the start of the path ($e_s$) is the entry node that an adversary uses to launch their attack, where multiple paths may exist between $e_s$ and $e_n$.

*Definition 3.2 (Event "path $\rho$ exists").* For a given path $\rho$, the event "path $\rho$ exists" is $\varepsilon_\rho = \{o | o \in \Omega \wedge \rho \subseteq o\}$.

Lemma 3.3. *For a path $\rho$, the probability of the corresponding path exists event $\varepsilon_\rho$ is*

$$P(\varepsilon_\rho) = \prod_{e_i \in \rho} P(e_i).$$

Proof. The probability of an event $\varepsilon_\rho$ is given by the probability of all the edges from $\varepsilon_\rho$, i.e., both those that are part of path $\rho$, and those that are not part of path $\rho$. In short $P(\varepsilon_\rho) = \prod_{e_i \in \rho} P(e_i) \cdot \prod_{\bar{e}_j \notin \rho} P(\bar{e}_j)$ where $e_i, \bar{e}_j \in \varepsilon_\rho$ are the edges that are part of $\varepsilon_\rho$, and not part of $\varepsilon_\rho$ respectively.

However, $\varepsilon_\rho$ contains every possible combination of edges $\bar{e}_j$ that are not part of $\rho$ meaning that $\prod_{\bar{e}_j \notin \rho} P(\bar{e}_j) = 1$. As such, $P(\varepsilon_\rho) = \prod_{e_i \in \rho} P(e_i)$. □

Lemma 3.4. *The probability of a set of events $\mathcal{P}$ described by the existence of n paths $\rho_i$ is computed as:*

$$\begin{aligned} P(\mathcal{P}) &= P(\bigcup_{i=1}^{n} \rho_i) \\ &= \sum_{k=1}^{n} (1)^{k+1} \cdot \sum_{\substack{\varepsilon_{\rho_i} \subseteq \mathcal{P} \\ |\varepsilon_{\rho_i}|=k}} P\left(\cap \varepsilon_{\rho_i}\right) \end{aligned} \tag{3}$$

Proof. Lemma 3.4 can be seen to be a probabilistic application of the inclusion-exclusion principle [4]. □

## 3.3 Tool Design

In this section, we present our implementation of the tool, highlighting some design choices and optimisations.

**Attack vector discovery.** Our tool models systems using directed graphs such that, for a system characterised by graph $G(N, E)$, the possible probabilistic attack vectors of this system diagram will be paths from $G(N, E)$. We are using the Breadth-first search algorithm [9] to discover these paths. The Breadth-first algorithm is a simple and efficient search algorithm with a $O|N|$ worst-case complexity, where $|N|$ represents number of nodes in the graph.

**Superimposing and routing.** System model diagrams are usually complex and contain multiple layers of information (i.e. a connection can contain details about the protocols used, the physical medium, the purpose of data, etc.) and performing accurate and meaningful security computations on these models is not trivial. For example, to obtain an accurate analysis of the system presented in Fig. 1 one would need to decompose the system into 12 independent graphs (i.e. two medium types: wired, wireless and 6 connection types), and apply the computation method outlined in Section 3.2 for each of these graphs. This method, however, while somewhat accurate would still not capture any interaction between these connections. We give a detailed example of this in Section 5.3.

As such, in our tool we use a layer-aware Breadth-first search algorithm which supports "network routing"-like functionality. What this means is that nodes are appended to a path only if they both support a specific *data type* and are connected by an edge that also supports that *data type*. In order to also capture the interaction between layers, we are using the *bridge* elements of the graph nodes. These describe the capability of a node to convert data from one layer (e.g. Wired connection, Wi-Fi connection) to another. A path can, thus, be comprised of several shorter paths, each belonging to a different layer, if they share bridge nodes that connect these layers. For example, in Fig. 2, the path `[GSM-R]->[GSM-R base]->[MSC]` is a valid path across wired (continuous line) and wireless (dotted line) connection types if the `[GSM-R base]` node is defined as a bridge between the two. The same path is also a valid across the *WAN network connection* (yellow line) and *LAN network connection* (green line) and if the `[GSM-R base]` node is configured as a bridge between them.

## 4 ANALYSIS PROFILES

The modelling tool presented in this paper enables system owners to carry out a number of tasks and analyses. The first and foremost type of analysis carried out by the tool is a discovery of all paths to key assets defined by the asset owner. In addition to path discovery, the tool is capable of carrying out alternative analyses including:

**'All Roads Lead to' analysis.** Assets in the graph can be considered 'bastions' which should be protected from attack. In our tool, paths which terminate at the specified assets are assessed, and a list returned of the paths that may be taken by an attacker which exceed some predefined threshold. As a concrete example, an RBC uses a number of direct sources of information before making a safety critical decision. However, indirectly-connected systems may have a more pronounced impact on the system than that of the directly connected systems.

**Patient-zero analysis.** For a given node which becomes compromised, it is important to understand the environmental effects that an attack has on a system. Given a specific starting node, the tool will follow paths from that node outwards, identifying paths that have a likelihood of being successfully compromised over a given threshold. This allows the asset owner to appreciate how an attack can propagate in ways that they had not identified.

**Testing new strategies.** ICS owners, given the architecture of their system, may wish to model and assess new strategies to better inform their security review process. Asset owners can use the tool to amend the CVSS vectors in the graph, or override the probability values to determine where changes are best made, and develop future strategies.

Core to the tool is the ability to discover paths, and carry out probabilistic analyses using the Inclusion-exclusion principle to assess how an attack may propagate throughout the model. This path discovery enables the asset owner to understand all the 'via' steps that an attacker may use as part of their cyber kill chain to compromise of one of the asset owner's identified critical assets.

## 5 APPLYING THE SCEPTICS TOOL TO ERTMS

In this section, we present our results from the tool assessing the ERTMS and train systems model presented in Figure 2. ERTMS is an example of a safety-critical ICS, formed of a number of interconnected systems, protocols and standards.

Our model represents a typical ERTMS and train architecture, where each part of the model can be as granular as required by the system owner. To demonstrate this, we have decomposed the train into its constituent components. As shown in Figure 2, we focus on individual flows, where one node represents a single unit, e.g.

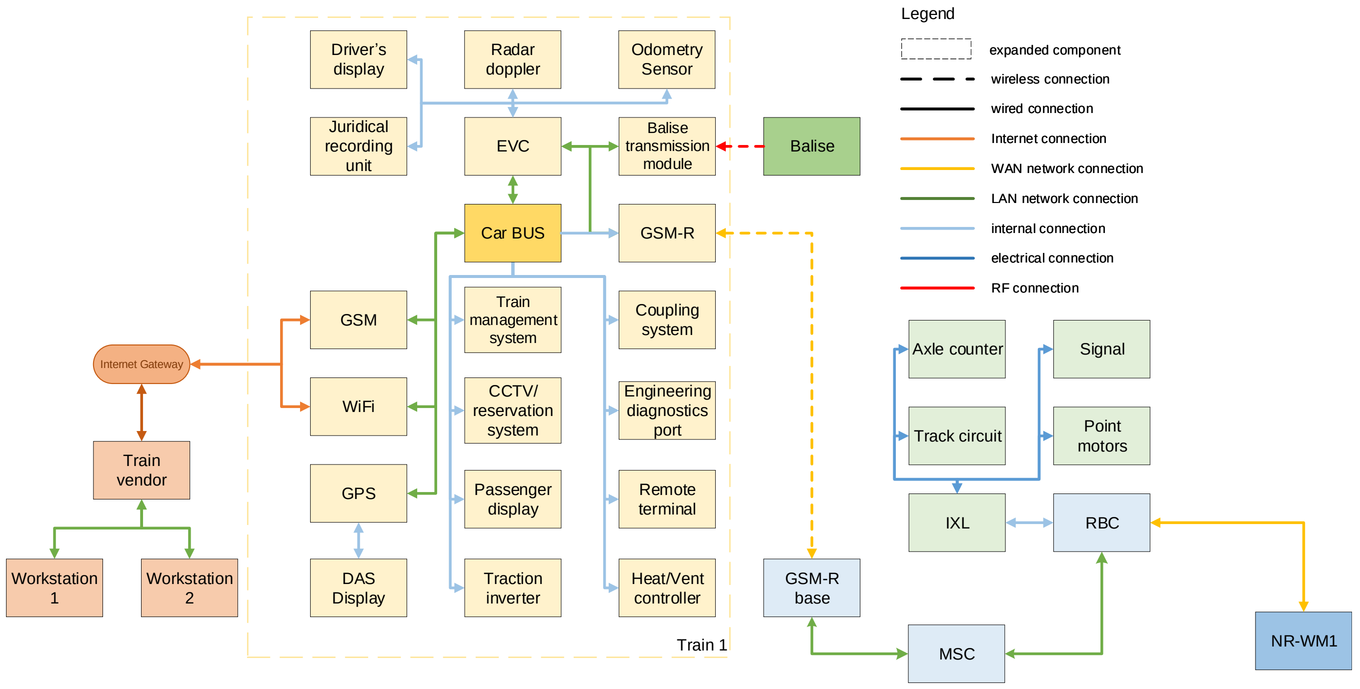

**Figure 2: Model of ERTMS, with a train expanded into its constituent components.**

a single passenger display. For each node and link in the model, a corresponding CVSS Profile (*CP*) was assigned, based on published work and interviews with rail experts. As input to the model, we specified a set of assets that the attacker would start their attack from, that is *Balise*, *Workstation1*, *Workstation2*, *GPS* and the *Car BUS*. For the set of assets that are to be assessed for exploitability, we specified the targets to be the *EVC*, *RBC*, *Car BUS*, *DAS Display* and *Passenger display*.

## 5.1 Selection of the Attacker Entry Points

For the set of attacker entry points (the adversarial model defined by the asset owner) above, we chose these specifically as they have differing interdependencies in the rail network. The Eurobalise is a RFID-like unit, fitted between the running rails and is trusted by the train to provide accurate location and track profile information[6]. The data presented by balise is used to provide accurate location data to the RBC such that it can make safety-critical decisions related to train positions. These decisions would then be communicated to the train using special messages called 'movement authorities'. In the event that a train reported an incorrect location, and there was no source to verify this, the train following would be given an overlapping movement authority. The train also relies on balise data for track profile information, e.g. tilting parameters, speed restrictions and gradients. If any of these were compromised, the train may be placed in an unsafe situation, as there is limited validation of this data.

[6]In ETCS Level 1, the balise is connected to a lineside unit which involves dynamic messages being sent to the train from the track. In Levels 2 and 3, the payloads are static and do not have a lineside unit. In ETCS Level 3, no trackside circuitry to verify train integrity and location is required

Remote Condition Monitoring is a diagnostic tool employed by train vendors to carry out real-time monitoring of train fleets. The data produced by the train and sent via this mechanism allows the vendor to plan preventative maintenance when the train returns to the depots, in addition to enabling remote triage of issues whilst the train is in service. As such, tampering with the data relayed to the vendor could lead to the service being withdrawn. Furthermore, if the train is remotely managed, the safe operation of the train could be affected, as an attacker could be able to carry out reconnaissance on the proprietary systems of the train. We explore the engineer workstations at the vendor as a possible entry point for the attacker in the model. In our model, we have two workstations, *Workstation1*, which has been compromised, and *Workstation2*, which is unaffected.

The on-board GPS currently is used only for supplementary services, e.g. providing location information to passenger information displays and for automated announcements. The effects of reporting a spoofed GPS location would be related to passenger disruption, e.g. clearing seat reservations or convincing passengers to depart the train at a station which was not their ultimate destination. In ERTMS, GPS does not form part of the safety-critical decision making process, however, outside of ERTMS, GPS is being used for the Driver Advisory System (DAS)[7], responsible for train and driver performance management, which we represent with the *DAS Display* component in the model. Where DAS is used, the train driver would see that the train is running ahead or behind schedule. This can result in the driver slowing the train, which would have a knock-on effect on following trains and route planning.

[7]https://www.rssb.co.uk/Library/groups-and-committees/2013-standalone-das-operational-concept.pdf

Finally, to highlight the increasing interconnected nature of systems onboard a train, we will examine the train bus network, analogous to the CAN Bus that exists in the automotive sector. Many train buses offer engineering diagnostic ports for an engineer to monitor the performance of a train or triage an issue whilst a train is still operating. We will assess the effect on the *Passenger Display*, *EVC* and *Train Management System*, a system available to the train manager and driver to control operational functions of the train (e.g. displaying diagnostic messages and 'authenticating' the driver of the train). The Car BUS presents a very real attack vector which may be used, as we have seen in the automotive sector, as discussed by Koscher et al. [18] and Checkoway et al. [6], where direct access via the OBD-II port on a car would allow the attacker to take full control of a car. We show a single bus for the train, which we believe to be a fair abstraction, where the model is taken to assume that there is a single bus where a multitude of devices are connected to. It is across this bus that all commands for the train are sent, where we reduce it into a single bus. [8]

## 5.2 Defining Target Assets

As target assets for the model, we chose systems which have either high safety requirements, or ones that, if compromised, would have significant disruptive effects. The European Vital Computer (EVC) is the train's on-board system for ERTMS, and controls train supervision. In the event that a balise provided inaccurate information, the EVC would pass this to the RBC, where an unsafe decision could be returned. The EVC also is responsible for on-board operations, where, if the balise reported an incorrect linespeed, there would be a risk to life.

The Car BUS is a bus that runs down the length of the train, and, in modern rolling stock, carries a variety of data, e.g. power and braking commands. In the event that data could be inserted here, the attacker would be able to control a number of train functions.

The DAS and passenger displays have no safety-requirement where they are considered only aids for a driver, and passengers respectively. They however have the ability to cause disruption to passengers and the rail network. The RBC, however, carries out safety-critical decisions, e.g. permitting trains to move forward for a set distance at a given speed. If this was compromised, trains could be given overlapping movement authorities or be put in a position of danger.

## 5.3 Results

We run the tool using the model shown in this paper, and with the assets and entry points as defined in this section, we obtain the following results:

```
Adversary a1:
[Balise]->[Balise transmission module]->[EVC]:0.32736
[Balise]->[Balise transmission module]->[EVC]->[GSM-R]->[GSM-R base]->[MSC]->
    ->[RBC]:0.00016
[Balise]->[Balise transmission module]->[EVC]->[GSM-R]->[GSM-R base]->[MSC]->
    ->[RBC]->[MSC]->[GSM-R base]->[GSM-R]->[EVC]:0.32747

Adversary a2:
[Workstation1]->[Train vendor]->[Internet Gateway]->[WiFi]->[Car BUS]:0.13456
[Workstation1]->[Train vendor]->[Internet Gateway]->[GSM]->[Car BUS]:0.03275
[Workstation2]->[Train vendor]->[Internet Gateway]->[WiFi]->[Car BUS]:0.0313
[Workstation2]->[Train vendor]->[Internet Gateway]->[GSM]->[Car BUS]:0.00762
```

[8]Some trains may separate this into individual buses, one per carriage, with another bus that runs across the length of the train.

```
Adversary a3:
[GPS]->[DAS Display]:0.97222
[GPS]->[Car BUS]->[Passenger display]: 0.9452
[GPS]->[Car BUS]->[EVC]->[GSM-R]->[GSM-R base]->[MSC]->[RBC]:0.00085

Adversary a4:
[Car BUS]->[EVC]:0.3805
[Car BUS]->[Passenger Display]:0.07178
[Car BUS]->[Train Management System]:0.07178
```

The results here are given as probabilities that the chain could be successfully exploited, that is the likelihood an attacker would be successful in breaching each component from the starting node to the 'critical asset'. A high probability would indicate that an attacker would be more likely than not to succeed, where high values should be used as a starting point for improvements by the asset owners. The attacks that are returned by the tool are likely and real attack vectors to the railway, confirmed by rail experts following a test of our tool, validating our results.

We observe that for Adversary *a1*, an attacker who has a Balise as an entry point, they have a probability of *0.32736* of successfully reaching and affecting the EVC on the train, for example issuing invalid line speeds, or convincing the train that it is in a different location. We note, however, that when we consider the attack affecting an RBC, the probability reduces to *0.00016*, but the return path, where the RBC has made a decision, has a much higher probability. This is because the balise location data flows to the EVC, and reported to the RBC, where the RBC will convert, i.e. bridge the data into command data, which flows back to the EVC. In existing tools, they would terminate analysis at the RBC, whereas our tool identifies this data conversion at the RBC, as the tool supports many datatypes for nodes and links, and is hence able to find this path.

For Adversary *a2*, however, the results are interesting – the compromised engineer workstation, *Workstation1* has a much higher probability of successfully affecting the Car BUS than a secure Workstation, *Workstation2*. In reality, if an engineer could remotely manage a train, this could be a potential threat, and highlights where good security practices are key to minimise exposure.

In the case of Adversary *a3*, we see how systems which have high reliance on the data offered by the GPS system, which has known attacks can affect systems on the train, but as the train does not use the GPS data for location reports to the RBC, the probability of it affecting the RBC is negligible, and this can be likened to an attack which uses the Car BUS to exploit some other vulnerability.

When we consider Adversary *a4*, an adversary who is onboard a train, where unauthorised access to the Car BUS would allow them to interface with a number of systems. The EVC has a, perhaps, higher likelihood of successful exploitation as the EVC directly interfaces with the Car BUS, where the EVC is responsible for maintaining safe operation of the train, thus would require the ability to carry out brake interventions. The probabilities given for the Passenger Display and Train Management System are the same, which can be explained to the same CVSS profile being applied to the links and that the Car BUS is mainly a means to transfer data across the system. For the Passenger Display, data may be transferred from the Train Management System (e.g. destination and passenger information), but has no significant role to play, hence the lower probability.

### 5.4 Testing New Strategies

From the results presented by the tool, the balise has a high probability of influencing the RBC, where a probability of $10^{-6}$ is considered the safety limit [29]. Using our tool, system owners can also simulate improvements to component security rather than just evaluating them. As an example, we replace the Balise in Fig. 2 with a new Balise, *SecureBalise*, capable of including a MAC to its payloads, we show how the simple addition of a MAC to the Balise payload prevents an attacker from setting their own payload without knowing the balise-derived key, reducing the path success likelihood from 0.32736 to 0.06591 for the path from the Balise to the EVC (shown below). This demonstrates how a system owner can improve the overall system security and reduce the probability of a successful attack. Another solution would be to send partial balise data to the RBC, which would carry out some validation. This would give a similar CVSS profile to the *SecureBalise* node, as the adversary would require a set of privileges and have an increased attack complexity, yielding similar probabilities to *SecureBalise*. Further alternative strategies, e.g. moving the balises to a setup similar to ETCS Level 1 (where the balise is connected to the RBC) can also be tested, but in reality would have significant costs.

```
[SecureBalise]->[Balise transmission module]->[EVC]:0.06591
[SecureBalise]->[Balise transmission module]->[EVC]->[GSM-R]->[GSM-R base]->
    ->[MSC]->[RBC]:0.00006
[SecureBalise]->[Balise transmission module]->[EVC]->[GSM-R]->[GSM-R base]->
    ->[MSC]->[RBC]->[MSC]->[GSM-R base]->[GSM-R]->[EVC]:0.06597
```

## 6 DISCUSSION AND FUTURE WORK

In this section, we discuss the benefits realised by the tool, its current shortcomings, and future development planned for the tool.

Where our tool leverages the CVSS framework to provide a contextual and verifiable means to calculate a probability of an asset becoming compromised, it is important to note that it represents a 'snapshot-in-time' for the system at the point of the model creation, developed as part of an iterative process to capture as many insights to make the model accurate. Over time, the model is expected to be revised as new assets are introduced, or mitigations have been made to reduce the exposure of high-risk assets.

Currently, the tool provides a step change for asset owners to assess the security of their infrastructure, as well as reasoning about the posterity and identify links that may not have been previously clear, or appreciated which may be leveraged by an attacker. It however still relies on the competency and capabilities of the asset owner to derive accurate and correct CVSS values, and also maintaining its currency. The tool provides a 'first-steps' approach to defining and assessing the security of infrastructure, and may be used to validate improvements made throughout the lifecycle of a system. Following discussions with Critical National Infrastructure operators and standards bodies, there are a number of extensions to the modelling tool we will consider as future work.

**Integration with existing architecture tooling.** Where our tool currently takes as its input a Visio diagram, some operators may use sector-specific software, where the transformation of these architectures might present a challenge to the asset owner. An evolution of our tool is into plugins to these sector-specific applications so that it may be run directly from a model, with an additional step of allowing visual tracing of attack paths.

**Taking costs of security into account.** Our tool allows asset owners to understand their infrastructure, and more importantly, identify potentially vulnerable systems that should be prioritised for further assessment to improve their posterity. These efforts, however, can incur significant expense. An extension to the tool would be to take into account the economics of improving security, where improvements can be suggested which are economically feasible, but provide a large return in reducing the attack space.

**Security Level/SIL segregation.** For system owners with a degree of maturity in Information Security, especially for critical infrastructure, there is a possible crossover between Security Level and Safety Integrity Level (SIL). Security levels provide a way for the system to be decomposed into 'zones' and 'conduits', where an agreed level is placed on them based on the complexity of an attack required to compromise that zone. SIL levels, however, apply to particular components, where the failure rate is based on the SIL level. Therefore, further extensions can be made to the tool to consider this zoning and partitioning of systems into 'islands' where the overall island should be considered.

In this paper, we take ERTMS and the UK rail network as a case study, breaking it down into its constituent components, where our assessments considered already-known weaknesses that exist in the ERTMS stack, but also highlighting subtle attack vectors which the rail industry has previously considered to not be an immediate issue. That said, the model we present in this paper is not exhaustive and can be made more granular, e.g. focusing on an 'exploded' component into more precise detail. Alternative ICS architectures can be modelled by the tool, however, due to the proprietary nature of other systems (for national security reasons), we chose to model ERTMS as it is an open standard, which allows us to thoroughly profile its security.

## 7 CONCLUSION

We have presented a modelling tool and methodology which enables ICS owners to assess the security of their infrastructure, and allow them to carry out 'first-steps' remediation. This is done through probabilistic analysis, path discovery and leveraging the CVSS framework to provide insights that are not possible in complex ICS architectures. The tool has further applications outside of the ICS sector, e.g. modelling corporate systems security and allows asset owners to reason about the security of their system with confidence, as well as providing assurances of the security in their systems.

## REFERENCES

[1] Ross J Anderson. 2010. *Security engineering: a guide to building dependable distributed systems*. John Wiley & Sons.
[2] Adam Beautement, Robert Coles, Jonathan Griffin, Christos Ioannidis, Brian Monahan, David Pym, Angela Sasse, and Mike Wonham. 2009. Modelling the human and technological costs and benefits of USB memory stick security. In *Managing Information Risk and the Economics of Security*. Springer, 141–163.
[3] Jens Braband. 1997. Safety and Security Requirements for an Advanced Train Control System. In *Safe Comp 97*. Springer, 111–122.
[4] Richard A. Brualdi. 2010. *Introductory Combinatorics (5th ed.)*. Prentice–Hall.
[5] Eric J. Byres, Matthew Franz, and Darrin Miller. 2004. The use of attack trees in assessing vulnerabilities in SCADA systems. In *in IEEE Conf. International Infrastructure Survivability Workshop (IISW '04). Institute for Electrical and Electronics Engineers*.
[6] Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, Stefan Savage, Karl Koscher, Alexei Czeskis, Franziska Roesner, and

Tadayoshi Kohno. 2011. Comprehensive Experimental Analyses of Automotive Attack Surfaces. In *Proceedings of the 20th USENIX Conference on Security (SEC'11)*. USENIX Association, Berkeley, CA, USA, 6–6. http://dl.acm.org/citation.cfm?id=2028067.2028073

[7] Pengsu Cheng, Lingyu Wang, Sushil Jajodia, and Anoop Singhal. 2012. Aggregating CVSS base scores for semantics-rich network security metrics. In *Reliable Distributed Systems (SRDS), 2012 IEEE 31st Symposium on*. IEEE, 31–40.

[8] Tom Chothia, Mihai Ordean, Joeri de Ruiter, and Richard J. Thomas. 2017. An Attack Against Message Authentication in the ERTMS Train to Trackside Communication Protocols. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security (ASIA CCS '17)*. ACM, New York, NY, USA, 743–756. https://doi.org/10.1145/3052973.3053027

[9] Thomas H Cormen, Charles E Leiserson, Ronald L Rivest, and Clifford Stein. 2001. Introduction to Algorithms Second Edition. (2001).

[10] Rhianne Evans, John Easton, and Clive Roberts. 2016. SCEPTICS: A Systematic Evaluation Process for Threats to Industrial Control Systems. (2016).

[11] Andrew Fielder, Tingting Li, and Chris Hankin. 2016. Modelling Cost-Effectiveness of Defenses in Industrial Control Systems. In *Computer Safety, Reliability, and Security*, Amund Skavhaug, Jérémie Guiochet, and Friedemann Bitsch (Eds.). Springer International Publishing, Cham, 187–200.

[12] Inc. FIRST.Org. 2016. Common Vulnerability Scoring System v3.0: Specification Document. (2016). https://www.first.org/cvss/cvss-v30-specification-v1.8.pdf

[13] Andrew Hawthorn. 2017. A Proven Approach to Requirements Engineering – The Why, What and How of REVEAL. (2017). https://conferences.ncl.ac.uk/media/sites/conferencewebsites/rssrail/Andrew%20Hawthorn%20-%20ALTRAN%20-%20A%20Proven%20Approach%20to%20Requirements%20Engineering.pdf

[14] Martin Hutle. 2015. SPARKS threat analysis using Attack Trees and Semantic Threat Graphs. (2015). https://hyrim.net/wp-content/uploads/2015/11/10_SPARKS_Threat_Analysis_Using_Attack_Trees_and_Semantic_Threat_Graphs.pdf

[15] Christos Ioannidis, David Pym, and Julian Williams. 2012. Information security trade-offs and optimal patching policies. *European Journal of Operational Research* 216, 2 (2012), 434–444.

[16] Rafiullah Khan, Kieran McLaughlin, David Laverty, and Sakir Sezer. 2017. STRIDE-based threat modeling for cyber-physical systems. In *Innovative Smart Grid Technologies Conference Europe (ISGT-Europe), 2017 IEEE PES*. IEEE, 1–6.

[17] Barbara Kordy, Sjouke Mauw, Saša Radomirović, and Patrick Schweitzer. 2011. Foundations of Attack–Defense Trees. In *Formal Aspects of Security and Trust*, Pierpaolo Degano, Sandro Etalle, and Joshua Guttman (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 80–95.

[18] Karl Koscher, Alexei Czeskis, Franziska Roesner, Shwetak Patel, Tadayoshi Kohno, Stephen Checkoway, Damon McCoy, Brian Kantor, Danny Anderson, Hovav Shacham, et al. 2010. Experimental security analysis of a modern automobile. In *Security and Privacy (SP), 2010 IEEE Symposium on*. IEEE, 447–462.

[19] Delphine Lautier and Franck Raynaud. 2013. Systemic Risk and Complex Systems: A Graph-Theory Analysis. In *Econophysics of Systemic Risk and Network Dynamics*. Springer, 19–37.

[20] Robert M. Lee, Michael J. Assante, and Tim Conway. 2016. Analysis of the cyber attack on the Ukrainian power grid. *Electricity Information Sharing and Analysis Center (E-ISAC)* (2016).

[21] Tingting Li and Chris Hankin. 2017. Effective Defence Against Zero-Day Exploits Using Bayesian Networks. In *Critical Information Infrastructures Security*, Grigore Havarneanu, Roberto Setola, Hypatia Nassopoulos, and Stephen Wolthusen (Eds.). Springer International Publishing, Cham, 123–136.

[22] Torsten Lodderstedt, David Basin, and Jürgen Doser. 2002. SecureUML: A UML-based modeling language for model-driven security. In *International Conference on the Unified Modeling Language*. Springer, 426–441.

[23] Jiqiang Lu, Zhen Li, and Matt Henricksen. 2015. Time-Memory Trade-off Attack on the GSM A5/1 Stream Cipher Using Commodity GPGPU. In *13th International Conference on Applied Cryptography and Network Security (ACNS 2015)*.

[24] Raimundas Matulevičius. 2017. *Fundamentals of Secure System Modelling*. Springer International Publishing, Cham. https://doi.org/10.1007/978-3-319-61717-6_2

[25] M. R. Orsman. 2012. The Victoria Line upgrade: From concept to full operation. In *IET Professional Development Course on Railway Signalling and Control Systems (RSCS 2012)*. 191–204. https://doi.org/10.1049/ic.2012.0052

[26] K. Prendergast. 2008. Condition monitoring on the Class 390 Pendolino. In *2008 4th IET International Conference on Railway Condition Monitoring*. 1–6. https://doi.org/10.1049/ic:20080311

[27] Riccardo Santini, Chiara Foglietta, and Stefano Panzieri. 2015. A graph-based evidence theory for assessing risk. In *Information Fusion (Fusion), 2015 18th International Conference on*. IEEE, 1467–1474.

[28] Adam Shostack. 2014. *Threat Modeling: Designing for Security* (1st ed.). Wiley Publishing.

[29] Jonathan Wolff. 2007. What is the Value of Preventing a Fatality? In *Risk: Philosophical Perspectives*, Tim Lewens (Ed.). Routledge.

## A CVSS EQUATIONS

The CVSS framework is an expressive means of describing security of systems. In our tool, we apply it to describe the security of a component or link. The vector is transformed into an overall score. In this appendix, we describe the CVSS framework and how values are determined.

The base score is calculated as a function of two sub-score equations, which relate to the impact and exploitability of a given system. The impact subscore (typically abbreviated as 'ISC') has two definitions, depending on whether the designer has identified whether the scope of the exploit changes as it progresses in the system, that is, if the scope does not change into a more 'privileged' position, the score is defined as $ISC = 6.42 \cdot ISC_{Base}$, where $ISC_{Base}$ is the impact base subscore, defined by Equation 5. In the case where the scope changes, the score is defined as:

$$ISC = 7.52 \times (ISC_{Base} - 0.029) - 3.25 \times (ISC_{Base} - 0.02)^{15} \tag{4}$$

$$\begin{aligned} ISC_{Base} = 1 - ((1 - Impact_{Confidentiality}) \\ \times (1 - Impact_{Integrity}) \times (1 - Impact_{Availability})) \end{aligned} \tag{5}$$

The exploitability subscore (ESC) is more straightforward, defined in Equation 6. The variables used in this equation and their numeric mappings are given in [12].

$$\begin{aligned} ESC = 8.22 \times AttackVector \times AttackComplexity \\ \times PrivilegeRequired \times UserInteraction \end{aligned} \tag{6}$$

In the event that the ISC metric is 0, then the base score is 0, otherwise, is given by Equation 7. The function $rup(x)$ rounds the value $x$ up by one decimal place, e.g. $rup(2.42) = 2.5$, $rup(2.9) = 2.9$.

$$BaseScore = \begin{cases} rup(min(1.08 \times (ISC+ESC), 10)), \textit{ IF scope changed} \\ rup(min((ISC+ESC), 10)), \textit{ IF scope unchanged} \end{cases} \tag{7}$$

## B ADVERSARY INPUT XML DEFINITION

```
<adversaries>
   <adversary id="a1">
     <entry_nodes>
       <!--AND operation-->
       <node>SecureBalise</node>
     </entry_nodes>
     <data_types>
       <type>any</type>
     </data_types>
       <link_types>
         <type>short-range wireless</type>
         <!--OR operation-->
         <type>long-range wireless</type>
       </link_types>
   </adversary>
</adversaries>
```

**Figure 3: Input XML Definition for an Adversary who has as their entry point a component in the model named *Secure-Balise*, and can use their capabilities on specific edge types.**

## C ASSET INPUT XML DEFINITION

```
<assets>
   <asset>
      <node>
         <id>RBC</id>
      </node>
      <data_types>
         <type>data</type>
      </data_types>
   </asset>
</assets>
```

**Figure 4: Input XML Definition for an Asset (*RBC*) for which is a target node, and subject to patient-zero and reachability analysis, with specific profiles and datatypes of interest.**